

Introduction

Serial Digital Interface (SDI) is a family of video interface standards from the Society of Motion Picture and Television Engineers (SMPTE) that transmits digital video data and embedded audio data through the 75-Ohm coaxial cable. SDI video standard is commonly used in professional video production facilities and television broadcasting systems.

HDMI is the most popular video/audio interface for transferring uncompressed video and compressed or uncompressed digital audio data in the consumer world.

This document describes a reference bridge design that allows users to seamlessly convert a broadcast quality SDI signal to HDMI signal with audio and HDMI signal to SDI with audio. The design works with the Lattice Tri-Rate SDI PHY IP core. The design has been validated on hardware using the Sparrowhawk FX development board with SDI add-on daughter card.

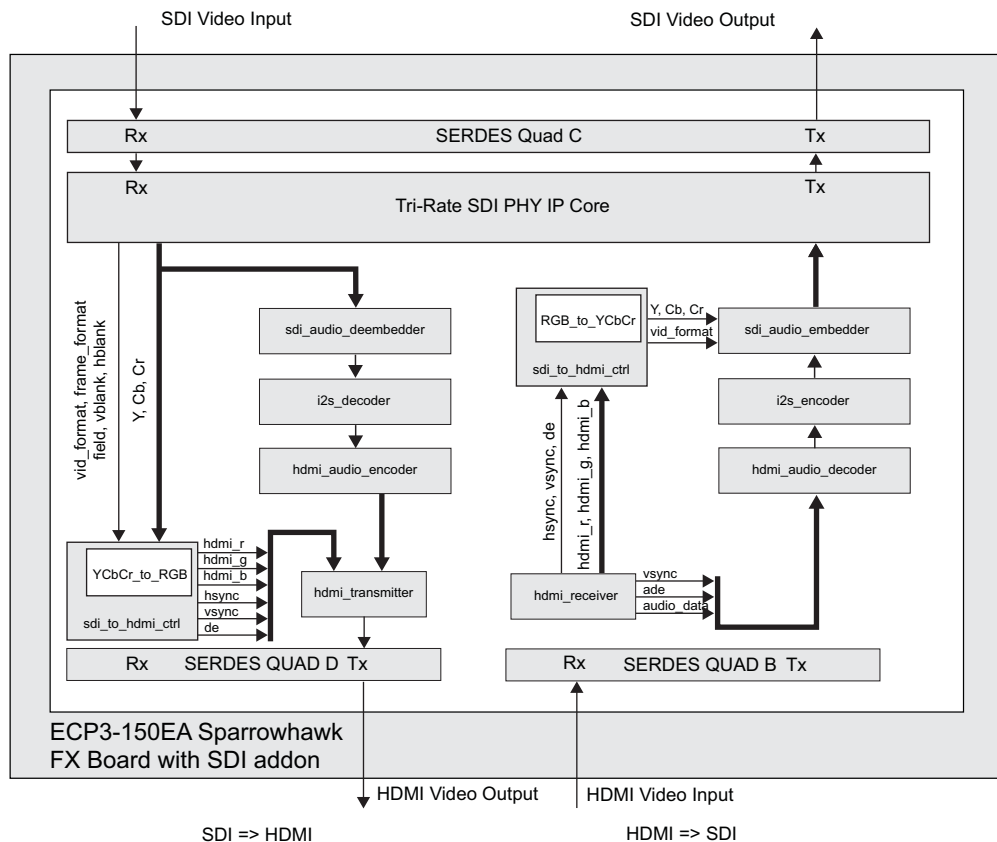
Features

- Support for the following video standards:
 - HD: SMPTE 274M, SMPTE 296M
 - 3G: SMPTE 425M (Level-A/B)
- Support for the following input/output video formats:
 - 1080p60
 - 1080p50
 - 1080p30
 - 1080p25
 - 1080p24
 - 720p60
- Automatic receive rate detection
- Lattice Mico32 soft core for configuration through I2C interface
- Support for I2S interface for linear PCM audio
- Support for sampling word width of 24 bits, sampling frequency of 48 kHz and two audio channels

Functional Description

The SDI-HDMI reference bridge design is logically divided into two portions: SDI to HDMI bridge and HDMI to SDI bridge. Each of these portions can also be divided into video and audio processing data path.

Figure 1. Block Diagram



SDI to HDMI

When converting SDI to HDMI video, the received data and timing control signals from the Tri-rate SDI PHY core are converted to HDMI interface signals in the module `sdi_to_hdmi_ctrl` which includes a YCbCr to RGB convertor. Then they are encoded by the module HDMI Transmitter which is connected to the SERDES transmitter.

The audio data is separated from the parallel video data received by the Tri-Rate SDI PHY IP core by the SDI audio de-embedder core, then the I2S stream is decoded into parallel data by the I2S decoder core and fed into the HDMI audio encoder.

HDMI to SDI

When converting HDMI to SDI video, the received data and timing control signals from HDMI receiver core are converted to SDI interface signals in the module `hdmi_to_sdi_ctrl` which includes a RGB to YCbCr convertor. Then they are processed by the Tri-rate SDI PHY IP core which is connected to the SERDES transmitter quad.

When converting HDMI to SDI audio, the received audio data signals from HDMI receiver core are decoded by the HDMI audio decoder core, encoded to I2S stream by the I2S encoder and then embedded in the SDI video stream by the SDI audio embedder core.

There is no frame buffer in the demo, so the resolution and the frame rate should be supported by both SDI and HDMI specification. Currently, the supported formats are 1080p60, 1080p50, 1080p30, 1080p25, 1080p24 and 720p60.

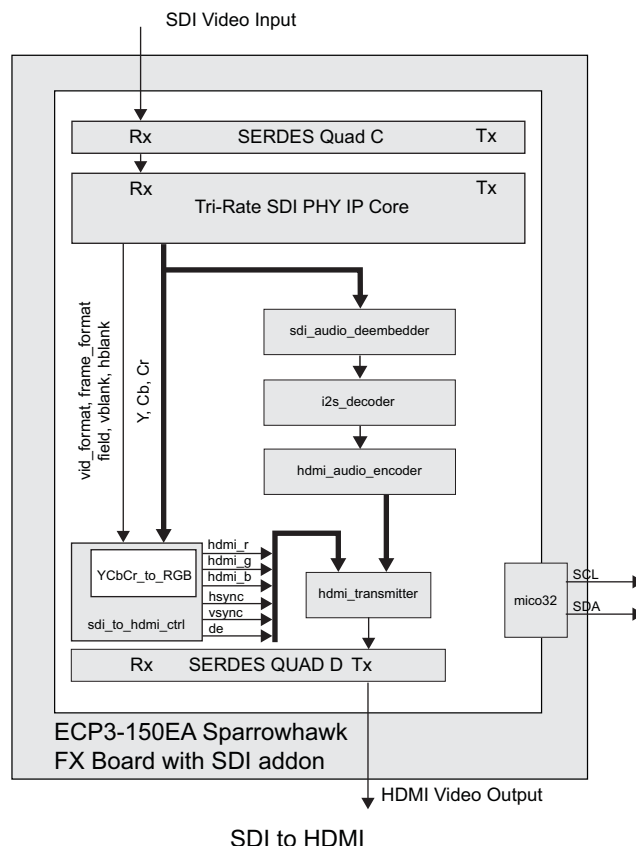
Table 1. Ports Definitions

Ports	Width	I/O	Description
wd_clk	1	In	100 MHz input clock
sdi_inp_ch1	1	In	SDI high-speed PCS input, positive, channel 1
sdi_inn_ch1	1	In	SDI high-speed PCS input, negative, channel 1
sdi_refclkp	1	In	SDI reference clock input, positive
sdi_refclkn	1	In	SDI reference clock input, negative
sdi_outp_ch1	1	Out	SDI high-speed PCS output, positive, channel 1
sdi_outn_ch1	1	Out	SDI high-speed PCS output, negative, channel 1
refclkp	1	In	HDMI reference clock input, positive
refclkn	1	In	HDMI reference clock input, negative
hdoutp0	1	Out	HDMI high-speed PCS output, positive, channel 0
hdoutn0	1	Out	HDMI high-speed PCS output, negative, channel 0
hdoutp1	1	Out	HDMI high-speed PCS output, positive, channel 1
hdoutn1	1	Out	HDMI high-speed PCS output, negative, channel 1
hdoutp2	1	Out	HDMI high-speed PCS output, positive, channel 2
hdoutn2	1	Out	HDMI high-speed PCS output, negative, channel 2
hdoutp3	1	Out	HDMI high-speed PCS output, positive, channel 3
hdoutn3	1	Out	HDMI high-speed PCS output, negative, channel 3
i2cm_ocSDA	1	In/Out	I2C data line
i2cm_ocSCL	1	In/Out	I2C clock line
hdmi_in_scl	1	In	DDC I2C clock line
hdmi_in_sda	1	In/Out	DDC I2C data line
hdmi_in_hpd	1	In/Out	DDC hot plug detect
hdmi_out_oe_n_0	1	Out	HDMI level shifter chip STHDLS101T TMDS output enable pin dedicated to the DVI connector J3 (PCSB SERDES quad)
hdmi_out_ddc_en_0	1	Out	HDMI level shifter chip STHDLS101T DDC and HPD enable pin dedicated to the DVI connector J3 (PCSB SERDES quad)
fpga_vsync	1	Out	Vertical sync signal of the received HDMI stream
fpga_hsync	1	Out	Horizontal sync signal of the received HDMI stream
clk_sel_0	1	Out	SDI reference clock select output, set high
clk_sel_1	1	Out	SDI reference clock select output, set high
sdi_sdhd_out0_1	1	Out	HD or SD status output
i2s_bclk	1	Out	I2S bit clock
i2s_wclk	1	Out	I2S word clock
i2s_din	1	Out	I2S output data, sampled at the rising edge of I2S bit clock
sw_dip	4	In	4 dip switches
sw_push	4	In	4 push buttons
led	8	Out	8 LED status outputs

Video Interface

SDI to HDMI

Figure 2. SDI to HDMI Data Path Block Diagram



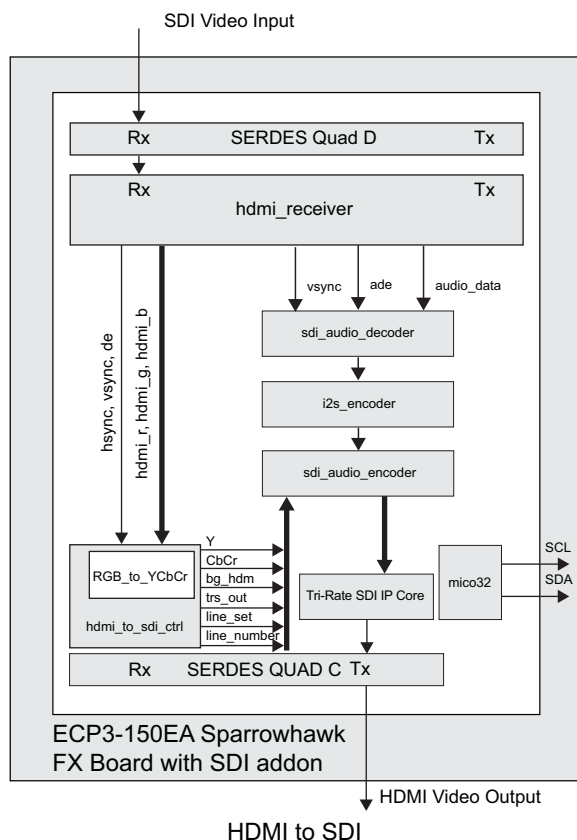
The SDI serial video stream is received by the LatticeECP3 SERDES PCS and descrambled and word-aligned by the Tri-Rate SDI PHY IP core receiver. The received parallel video data is then sent to sdi_to_hdmi_ctrl core that generates the HDMI timing reference signals and converts the YCbCr pixel data received from SDI to RGB values. The HDMI Transmitter core then encodes the video pixel data, audio and auxiliary data, hsync/vsync and other control signals into three 10-bit TMDS data. LatticeECP3 SERDES serializes the data for high-speed transmission. An additional SERDES channel is used to transmit the pixel clock.

Table 2. Ports Definitions for the Module *sdi_to_hdmi_ctrl*

Ports	Width	I/O	Description
rstn	1	In	Asynchronous reset, active low
clk	1	In	Input clock synchronous to input data control signals
vblank	1	In	Vertical blanking signal
hblank	1	In	Horizontal blanking signal
vid_format	2	In	Video format indicator (01 – 1280 x 720, 11 – 1920 x 1080)
frame_format	3	In	Frame rate indicator
data_in	20	In	Parallel video data input
line_number	11	In	Line number
interlace	1	Out	Interlaced video indicator, active high
hsync	1	Out	Hsync signal
vsync	1	Out	Vsync signal
de	1	Out	Data enable signal
hdmi_r	8	Out	8-bit video data red component
hdmi_g	8	Out	8-bit video data green component
hdmi_b	8	Out	8-bit video data blue component

HDMI to SDI

Figure 3. HDMI to SDI Data Path Block Diagram



The HDMI serial data streams are recovered and de-serialized over the three SERDES receive channels with the quad reference clock driven by the HDMI pixel clock. The embedded PCS Word Aligner detects the four Control Character patterns in the serial data stream and aligns the 10-bit TMDS character boundary before transmitting the data to the FPGA fabric for decoding. The HDMI Receiver core decodes HDMI data stream by automatically detecting the data format and decoding the data with the proper decoders. The decoded data is sent to the `hdmi_to_sdi_ctrl` core that generates the SDI timing reference signals and converts the RGB pixel data received from HDMI to YCbCr values. Parallel video data is then sent to the SDI audio embedder core which outputs the parallel video data with/without the embedded audio data and timing reference sequence identifier, both of which are connected to the SDI PHY IP core for data scrambling, line number insertion and CRC insertion before being sent out.

Table 3. Ports Definitions for the Module `hdmi_to_sdi_ctrl`

Ports	Width	I/O	Description
<code>rstn</code>	1	In	Asynchronous reset, active low
<code>clk</code>	1	In	Receive channel recovered clock
<code>wd_clk</code>	1	In	100 MHz clock input
<code>hsync</code>	1	In	HSYNC video control input signal
<code>vsync</code>	1	In	VSYNC video control input signal
<code>de</code>	1	In	Video data enable signal
<code>hdmi_r</code>	8	In	8-bit TMDS decoded video data of Channel 2
<code>hdmi_g</code>	8	In	8-bit TMDS decoded video data of Channel 1
<code>hdmi_b</code>	8	In	8-bit TMDS decoded video data of Channel 0
<code>tg_hdn</code>	1	Out	3G/HD indicator. 1: 3G; 0: HD/SD
<code>video_int</code>	1	Out	Interlaced video frame indicator. 1: interlaced; 0: progressive
<code>frame_format</code>	3	Out	Frame rate indicator.
<code>vic</code>	7	Out	Video input change indicator
<code>line_number</code>	11	Out	Line number output
<code>line_total</code>	11	Out	Total number of lines
<code>line_set</code>	1	Out	
<code>trs_out</code>	1	Out	Output TRS signal
<code>sdi_data</code>	20	Out	Parallel video data output

EDID

Mikroprojekt's `iq_i2c_rom` core (.ngo netlist file) is used in the design for storing the extended display identification data (EDID) to describe supported resolution and timing to a HDMI video source.

The video source can read out the EDID over the DVI connector DDC lines (SDA and SCL).

Audio Interface

SDI to HDMI

SDI Audio De-embedder

For the SDI Audio Data De-embedder, the audio data is separated from the parallel video data out of the Tri-Rate SDI PHY IP core. Then the parallel audio data is encoded to I2S. The status signals that indicate the CRC error and ECC error, the delay value that point out the audio processing delay relative to video and the audio information that includes the Z/C/U/V bits are also extracted.

Table 4. Ports Definitions for the Module *sdi_audio_de_embedder*

Ports	Width	I/O	Description
rstn	1	In	Asynchronous reset, active low
pdo_clk	1	In	Receiver pixel clock, synchronous with pd_out
clk_74_25	1	In	Local generated clock with the frequency 74.25MHz, could also be other frequencies with no less than 74.25MHz
rx_hd_sdn	1	In	SD/HD indicator, 1: HD/3G; 0: SD
rx_tg_hdn	1	In	3G/HD indicator, 1: 3G; 0: HD/SD
rx_lb_lan	1	In	3G Level-B indicator, 1: 3G Level-B; 0: HD/SD/3G Level-A
trs_out	1	In	From SDI core receiver. This output is high during the start of the TRS sequence
pd_out	20	In	Parallel data from SDI core receiver with embedded audio data
aes3_bp_clk	1	In	AES3 transmitter bi-phase bit clock
i2s_bclk_out	1	In	I2S bit clock
i2s_lrcclk_out	1	In	I2S channel indicator. 0: left channel; 1: right channel
rxlb_stream	1	In	Link selection(location of audio source) for 3G Level-B, 0: Link-A(pd_out[19:10]); 1: Link-B(pd_out[9:0])
i2s_dout	CH_NUM_RX	Out	I2S data, sampled at the rising edge of I2S bit clock
aes3_dout	CH_NUM_RX	Out	AES3 serial output data
audio_info_out	CH_NUM_RX*2	Out	Extracted Audio Z/C/U/V bits
ecc_err	CH_NUM_RX/2	Out	ECC error indication. 1: error; 0: no error
crc_err	CH_NUM_RX/2	Out	CRC error indication. 1: error; 0: no error
del_out	CH_NUM_RX*26	Out	Audio processing delay relative to video, extracted from the audio control packet, usually a constant value when a system is stable, so it can be sampled by any clock

The parameter CH_NUM_RX can be set to 2, 4 or 8 in. This is used to indicate the number of stereo channels. For example, if it is set to four, four stereo channels (four left and four right sub-channels) which form two audio groups (each with two stereo channels) will be received. The channel number parameter in the SDI to HDMI reference design is set to 2.

I2S Interface

I2S is a three-wire serial digital audio protocol. One I2S interface contains two audio channels. There are three possible audio data formats for an I2S interface: I2S format, Left Justified and Right Justified. Only the I2S format is supported in this reference design.

Table 5. Ports Definitions for the Module *i2s_decoder*

Ports	Width	I/O	Description
rstn	1	In	Asynchronous reset, active low
sample_width_sel	2	In	I2S sampling word width selection, 00/11: 24bit, 01: 20bit, 10: 16bit
i2s_bclk_in	1	In	I2S bit clock
i2s_lrcclk_in	1	In	I2S channel indicator. 0: left channel; 1: right channel
i2s_din	1	In	I2S input data, sampled at the rising edge of I2S bit clock
audio_data_l	24	Out	Decoded data from the left channel, connected with the module <i>hdmi_audio_enc</i>
audio_data_r	24	Out	Decoded data from the right channel, connected with the module <i>hdmi_audio_enc</i>

HDMI Audio Encoder

The HDMI audio encoder core uses TMDS Error Reduction Coding (TERC4) to encode 4-bit audio and auxiliary data into the 10-bit TMDS sequence during the Data Island Period.

Table 6. Ports Definitions for the Module *hdmi_audio_encoder*

Ports	Width	I/O	Description
pix_clk	1	In	Transmit pixel clock
rstn	1	In	Asynchronous reset, active low
osc_in	1	In	27 MHz clock input
hsync	1	In	HSYNC video control signal for transmit
vsync	1	In	VSYSNC video control signal for transmit
de	1	In	HDMI video data enable for transmit
vic	7	In	
aud_cts	20	In	Cycle time stamp (denominator of the ratio of TMDS clock to audio clock), transmitted in ACR packet
aud_n	20	In	Numerator of the ratio of TMDS clock to audio clock, transmitted in ACR packet
aud_clk	1	In	Audio clock
aud_wr	1	In	Audio data valid
aud_left	24	In	Audio left channel data
aud_right	24	In	Audio right channel data
tx_ade	1	Out	Audio/Aux Data Enable signal
tx_aux_ch0	4	Out	4-bit audio/aux data of Channel 0 for TERC4 encoding
tx_aux_ch1	4	Out	4-bit audio/aux data of Channel 1 for TERC4 encoding
tx_aux_ch2	4	Out	4-bit audio/aux data of Channel 2 for TERC4 encoding

HDMI to SDI

HDMI Audio Decoder

Table 7. Ports Definitions for the Module *hdmi_audio_decoder*

Ports	Width	I/O	Description
rstn	1	In	Asynchronous reset, active low
osc_in	1	In	100 MHz clock input
clk_in	1	In	Receiver pixel clock
rx_vsync	1	In	Receiver VSYNC video control signal
rx_ade	1	In	HDMI audio/auxiliary data enable for receiver
rx_audio_ch0	4	In	4-bit TERC4 decoded audio data of Channel 0
rx_audio_ch1	4	In	4-bit TERC4 decoded audio data of Channel 1
rx_audio_ch2	4	In	4-bit TERC4 decoded audio data of Channel 2
aud_det	1	Out	Audio Data Enable signal
aud_left	24	Out	Audio left channel data
aud_right	24	Out	Audio right channel data

I2S Interface

I2S is a three-wire serial digital audio protocol. One I2S interface contains two audio channels. There are three possible audio data formats for an I2S interface: I2S format, Left Justified and Right Justified. Only the I2S format is supported in these reference designs.

Table 8. Ports Definitions for the Module *i2s_enc*

Ports	Width	I/O	Description
rstn	1	in	Asynchronous reset, active low
clk_in	1	in	I2S bit clock
out_24bit	1	in	I2S output data width; 0: 32-bit, 1: 24bit
ch_left	24	in	Left channel parallel audio data, connected from <i>hdmi_audio_decoder</i>
ch_right	24	in	Right channel parallel audio data, connected from <i>hdmi_audio_decoder</i>
i2s_lrcclk_out	1	in	I2S channel indicator. 0: left channel; 1: right channel
i2s_dout	1	out	I2S output data, sampled at the rising edge of I2S bit clock, connected to module <i>sdi_audio_embedder</i>

SDI Audio Embedder

For the SDI Audio Data Embedder the audio data in I2S format is first decoded to parallel audio data, and then embedded in the SDI video stream before the stream is fed to the SDI PHY IP core. The SDI audio embedder also supports AES3 format audio data which is not implemented in this design.

Table 9. Ports Definitions for the Module *sdi_audio_embedder*

Ports	Width	I/O	Description
rstn	1	In	Asynchronous reset, active low
clk	1	In	Transmit pixel clock, synchronous with sdi_data_in
i2s_bclk_in	CH_NUM_TX	In	I2S bit clock
i2s_lrcclk_in	CH_NUM_TX	In	I2S channel indicator. 0: left channel; 1: right channel
i2s_din	CH_NUM_TX	In	I2S data, sampled at the rising edge of I2S bit clock
sys_clk	1	In	System clock, more than 4 x frequency of the AES3 bit clock
aes3_din	CH_NUM_TX	In	AES3 serial input data
sdi_data_in	20	In	Input SDI video data
hd_sdn_in	1	In	SD/HD indicator. 1: HD/3G; 0: SD Hardcoded to 1.
tg_hdn_in	1	In	3G/HD indicator. 1: 3G; 0: HD/SD Connected from hdmi_to_sdi_ctrl core
lb_lan_in	1	In	3G Level-B indicator. 1: 3G Level-B; 0: HD/SD/3G Level-A Hardcoded to 0
sd_525	1	In	SD 525 lines indicator. 1: means the video format is SD and the number of total lines is 525 in one frame; 0: other formats. Hardcoded to 0.
sd_625	1	In	SD 625 lines indicator. 1: means the video format is SD and the number of total lines is 625 in one frame; 0: other formats. Hardcoded to 0.
line_720	1	In	SDI 720 lines indicator. 1: means the number of active lines is 720 in one frame; 0: other formats Connected from hdmi_to_sdi_ctrl core
frame_rate	3	In	Frame rate indicator. 000: 60 Hz; 001: 50 Hz; 010: 30 Hz; 011: 25 Hz; 111: 24 Hz Connected from hdmi_to_sdi_ctrl core
progressive	1	In	Progressive video frame indicator. 1: progressive; 0: interlaced Connected from hdmi_to_sdi_ctrl core
del_in	CH_NUM_TX*26	In	Audio processing delay relative to video, inserted to the audio control packet
sample_width_sel	2	In	I2S sampling word width selection, 00/11: 24bit, 01: 20bit, 10: 16bit
audio_info_in	CH_NUM_TX*2	In	Audio Z/C/U/V bits from external ports, which is depending on the parameter definition, should better be provided at the falling edge of audio sampling clock
pd_in	20	Out	Parallel video data output with embedded audio data
trs_in	1	Out	Output TRS signal
vpid1_in	32	Out	Video payload identifier output to SDI IP. It is not used when transmitting SD video. It is connected with vpid1_in of SDI IP.
vpid2_in	32	Out	Video payload identifier output to SDI IP. Only valid for stream 2 for 3G level-B video

The parameter CH_NUM_TX can be set to 2, 4 or 8. This is used to indicate the number of stereo channels. For example, if it is set to four, four stereo channels (four left and four right sub-channels) which form two audio groups (each with two stereo channels) will be transmitted/received. The channel number parameter in the HDMI to SDI bridge design is set to 2.

Mico32

Mico32 embedded microcontroller is used for configuring the LMH1982 video clock generator and TLV320AIC stereo audio codec on the SDI add-on board through I2C interface. Mico32 automatically detects through GPIO interface if a change of mode (3G, HD) occurs and reconfigures the LMH1982 chip accordingly.

Clocking Requirements

Table 10. Clocking Requirements for SDI-HDMI Reference Bridge Design

Clocks	Description	Frequency
wd_clk	100 MHz clock oscillator input to the FPGA	100 MHz
refclk	HDMI quad reference clock input	148.5 MHz
rx_cref_clk	Locally generated clock with the frequency of 148.5 MHz. Used in SDI quad as receive reference clock from FPGA logic	148.5 MHz
clk_74_25	Locally generated clock with the frequency of 74.25 MHz.	74.25 MHz
pdo_clk	Parallel data output clock from SDI PHY IP. This clock is a multiplexed version of rx_clk and rx_full_clk. If this clock is available, the output data as well as output status and control signals are synchronous with this clock.	3G: 148.5 MHz HD: 74.25 MHz
rx_half_clk	SDI quad receive channel recovered half clock.	
txfullclk	HDMI quad TX PLL full rate clock	
hdmi_tx_refclk	HDMI quad TX reference clock from FPGA logic (same as rx_half_clk)	
sdi_refclk	SDI quad reference clock input	3G: 148.5 MHz HD: 74.25 MHz
tx_half_clk	SDI quad tx pll half clock	
refclk2fpga	HDMI quad reference clock to FPGA core	
rxfullclk	HDMI quad receive channel recovered clock	3G: 148.5 MHz HD: 74.25 MHz
clk_96m	Locally generated clock with the frequency of 96 MHz.	96 MHz
clk_23m	Locally generated clock with the frequency of 23 MHz.	23 MHz
clk_27m	Locally generated clock with the frequency of 27 MHz.	27 MHz
i2s_bclk	I2S bit clock	
i2s_wclk	I2S word clock	

Sparrowhawk FX Implementation

The SDI DVR reference design demonstration runs on Sparrowhawk FX board with SDI add-on connected to one or more SDI video input sources and HDMI video output.

Prerequisites

The hardware, software, cable, general requirements and setup procedure for this demonstration are given in the following sections.

Hardware Requirements

This demonstration requires the following hardware components:

- Sparrowhawk FX board with SDI add-on
- 12 V DC power supply
- PC

Software Requirements

This demonstration requires the following software components:

- Lattice Diamond® Programmer tool
- ORCAstra software for user control interface - optional

Cable Requirements

This demonstration requires the following cable components:

- 2x SDI (BNC to BNC) video cables for input and output
- 2x DVI or DVI/HDMI video cable for input and output
- Lattice JTAG to USB download cable

Setup Procedure

1. Check prerequisites.
2. Setup hardware.
3. Download FPGA bitstream.
4. Run the demo.

Limitations

The bitstream included with the demo design has a time-out restriction. It will only allow evaluation operation for about four hours because the Tri-Rate SDI Phy IP core evaluation version contains a built-in timer. A licence is required to generate bitstreams that do not include the hardware evaluation timeout limitation.

Hardware Setup

Important parts for the setup are outlined in red on the Sparrowhawk FX board layout.

- **JTAG** – used for download of FPGA designs. Connect Lattice JTAG cable here when downloading.
- **Power connector and switch** – used to power the board.
- **DVI Out #0** – mixed video output. Connect to the DVI connector of the monitor.

Figure 4. Sparrowhawk FX Board

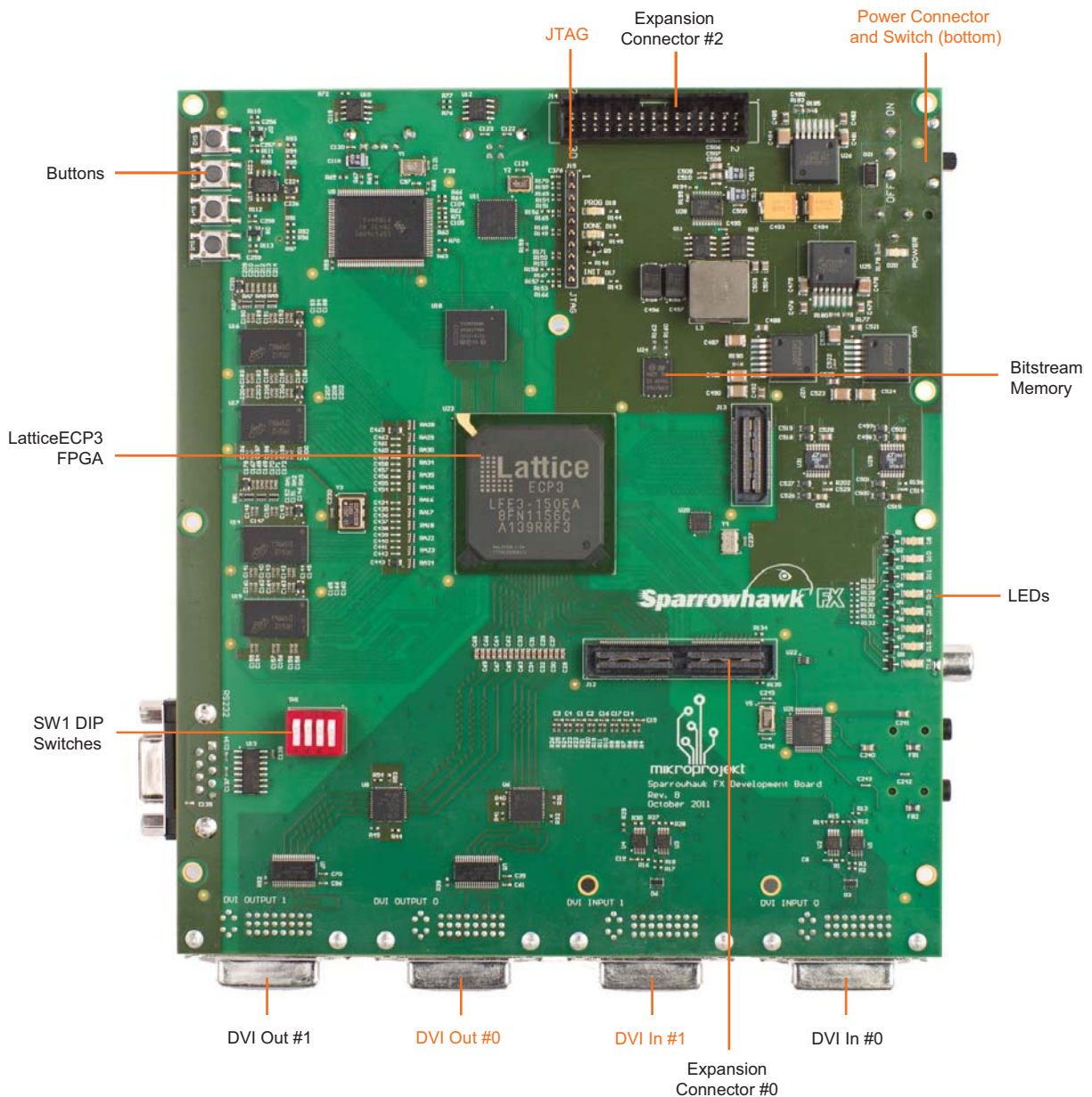


Figure 5. SDI Addon Board

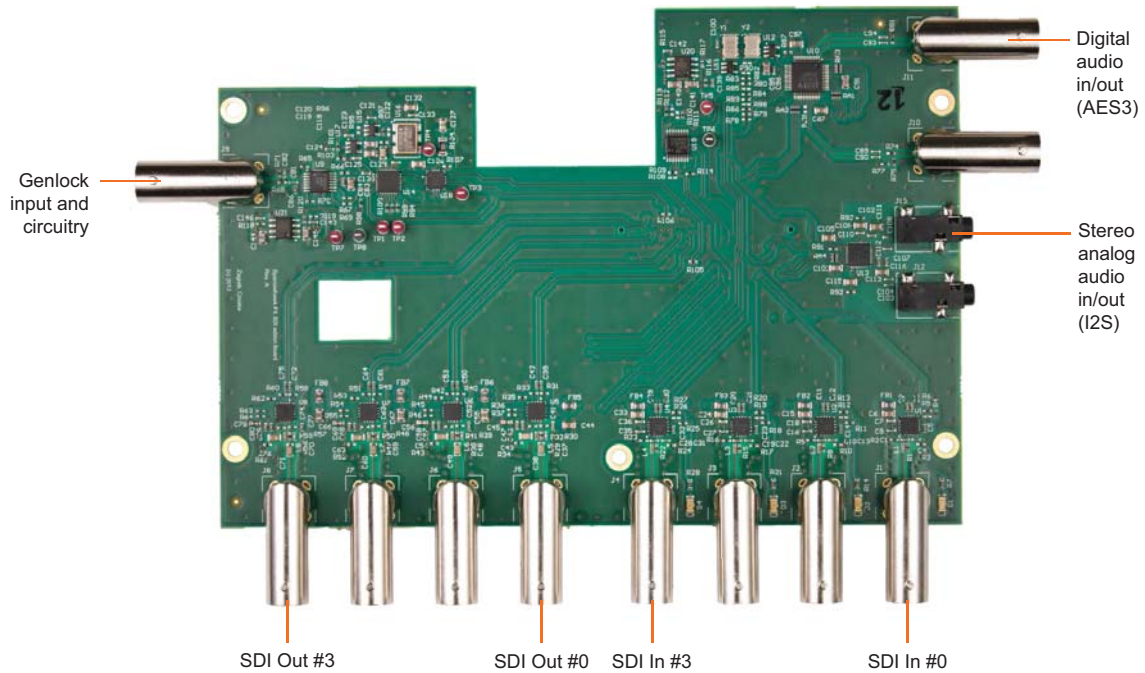
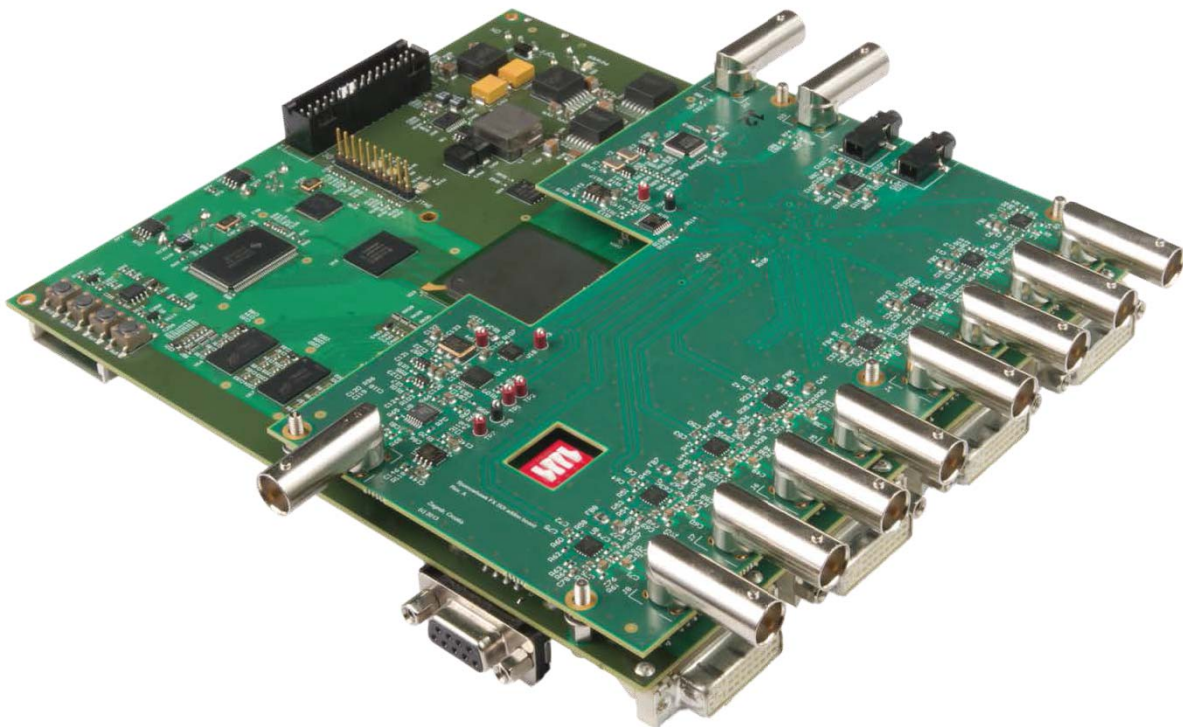


Figure 6. Sparrowhawk FX with Mounted SDI Addon

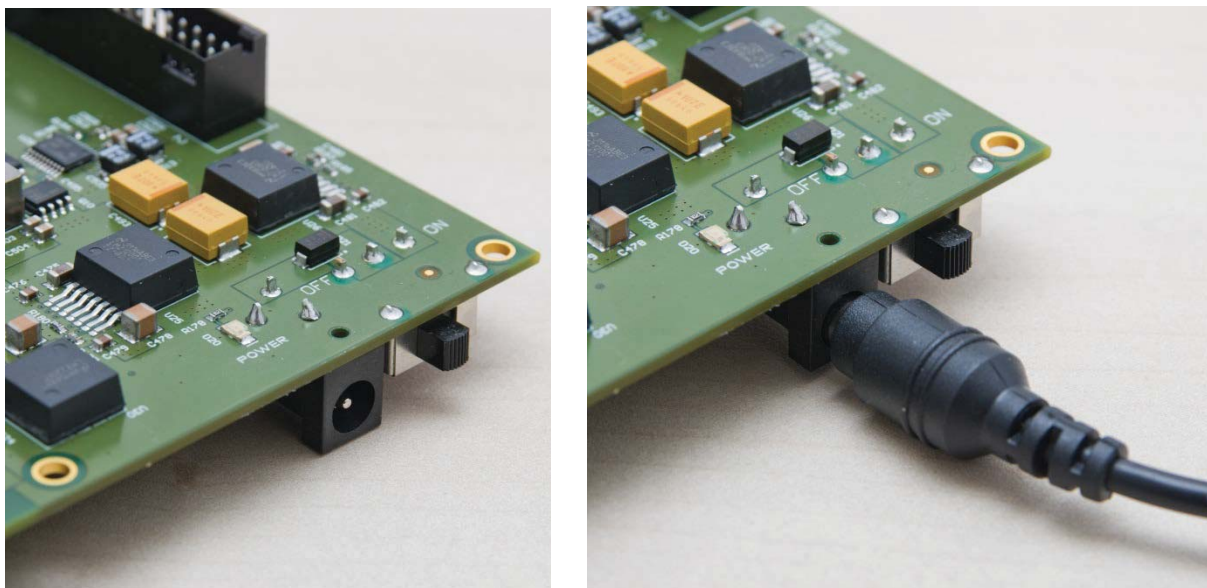


Powering Up the Board

The Sparrowhawk FX with SDI add-on can be powered by the AC transformer provided with the board, or any type of DC supply source, providing 12 V DC and a minimum of 18 W.

The 12V DC power supply should be connected to the connector J16 on the bottom side of the board. The Sparrowhawk FX is protected by the diode D12 from the reverse power connection. The board is turned on/off by toggling the switch SW6, with *ON* and *OFF* marked in the silkscreen on the top of the board.

Figure 7. Sparrowhawk FX 12 V DC Power Connection



Connecting Video Sources and Sinks

The SDI and DVI video sources and sinks should be connected as shown in the following figures.

HDMI video input should be connected to the input #1. The received HDMI video is converted to SDI signaling and output to SDI output #0, as shown in Figure 8.

SDI video input should be connected to the input #0. The received SDI video is converted to HDMI signaling and output to DVI output #0, as shown in Figure 9.

SDI and HDMI video inputs support 1920 x 1080 or 1280 x 720 input resolution.

Figure 8. DVI Input, SDI Output



Figure 9. SDI Input, DVI Output

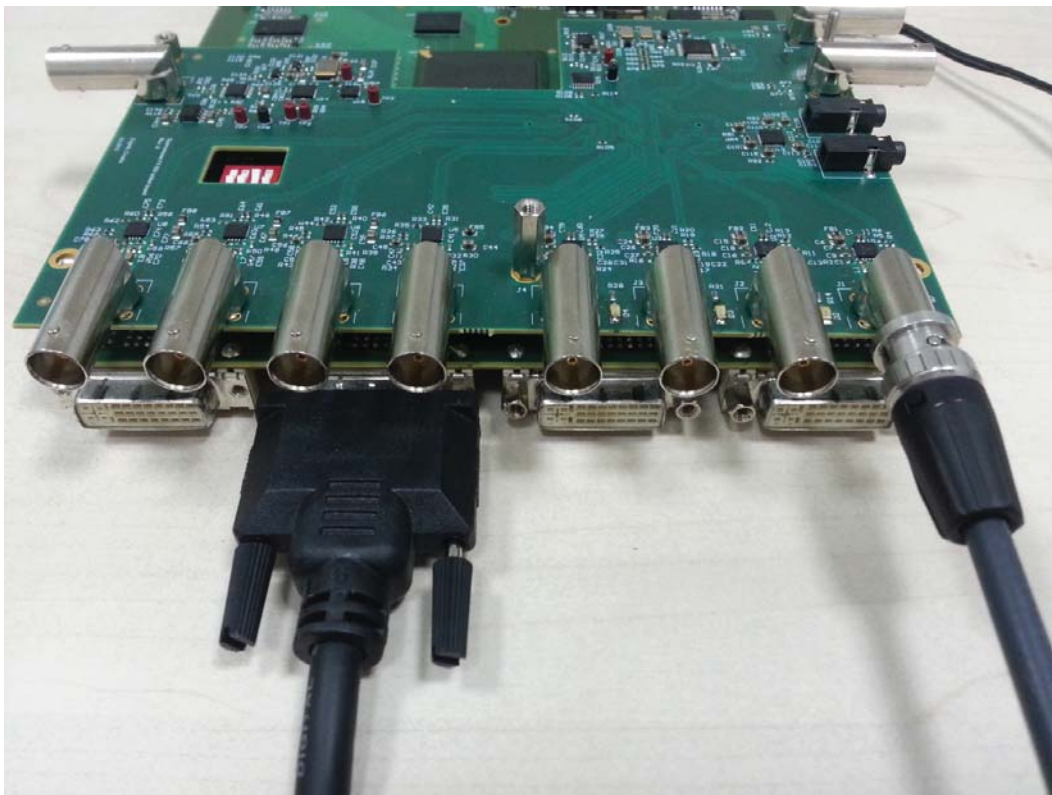


Figure 10. DVI Input, SDI Output to SDI Input Loopback, HDMI Output



Pinout Maps and Hardware Assignments

Push button SW5 can be used to reset the SDI PCS.

Figure 11. SW Push Button for Resetting the SDI PCS

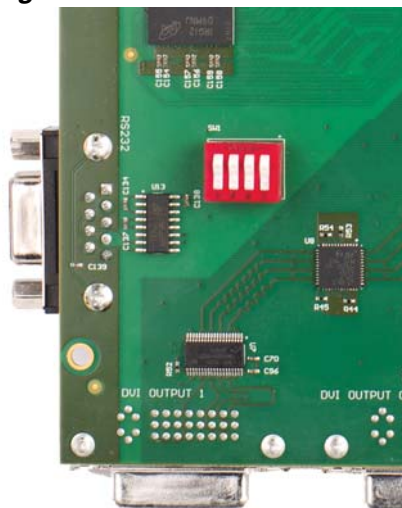


Table 11. Orcastra and Push Button Reset PCS Mapping

FPGA LatticeECP3 PCS	Orcastra	Function	Connector	Reset
PCSB	ECP3 PCS 3	HDMI RX	DVI INPUT 1 on SHFX board	—
PCSC	ECP3 PCS 2 channel 1	SDI RX AND TX	J1 RX, J5 TX on SDI add-on	SW5
PCSD	ECP3 PCS 1	HDMI TX	DVI OUTPUT 0 on SHFX board	—

There are eight status LEDs on Sparrowhawk FX board that indicate the SDI channel status.

Table 12. LED Assignments

LEDs	Description	Silkscreen
led[0]	HDMI TX PCSB loss of lock	D9
led[1]	HDMI RX PCSD loss of lock	D10
led[2]	SDI TX tg_hdn	D11
led[3]	SDI RX tg_hdn	D12
led[4]	SDI TX pll_lol	D13
led[5]	SDI RX los_low	D14
led[6]	SDI_RX cdr_lol	D15
led[7]	SDI vid_active	D16

The device pinout is summarized in Table 13.

Table 13. Pinout

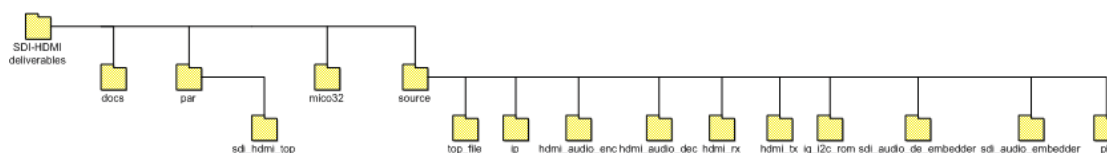
Ports	I/O	Lattice ECP3 BGA Ball	SysIO Bank
wd_clk	In	U6	7
i2cm_ocSDA	Inout	AN32	3
i2cm_ocSCL	Inout	AN31	3
hdmi_in_scl	In	W26	3
hdmi_in_sda	Inout	W27	3
hdmi_in_hpd	Inout	V26	3
hdmi_out_oe_n_0	Out	Y25	3
hdmi_out_ddc_en_0	Out	Y26	3
fpga_vsync	Out	AB30	3
fpga_hsync	Out	AC30	3
clk_sel_0	Out	AM32	3
clk_sel_1	Out	AM30	3
sdi_sdhd_out0_1	Out	AP33	3
i2s_bclk	Out	AF32	3
i2s_wclk	Out	AF34	3
i2s_din	Out	AG34	3
sw_dip[0]	In	A33	8
sw_dip[1]	In	A32	8
sw_dip[2]	In	B32	8
sw_dip[3]	In	C32	8
sw_push[0]	In	E11	0
sw_push[1]	In	E10	0
sw_push[2]	In	F10	0

Ports	I/O	Lattice ECP3 BGA Ball	SysIO Bank
sw_push[3]	In	F12	0
led[0]	Out	G23	1
led[1]	Out	H23	1
led[2]	Out	H22	1
led[3]	Out	G21	1
led[4]	Out	G26	1
led[5]	Out	H26	1
led[6]	Out	G25	1
led[7]	Out	H25	1

Reference Design Demo

List of Files

Figure 12. Directory Structure of the SDI_DVR Deliverables



Description of directories in the deliverables folder (not all directories and the belonging files are listed here; this description is for orientation only):

- \docs: contains the following PDF documents delivered with the SDI-HDMI bridge reference design
 - SDI-HDMI user guide (this document)
 - Sparrowhawk FX user's manual
 - SDI addon board user's manual
- \par : Diamond project directory, contains an .ldf Diamond design project file, .lpf constraint file, all netlist NGO files required for the FPGA design, and bitstream file containing FPGA design (in the \par\sdi_hdmi_top subdirectory).
- \mico32: mico32 soft core directory
- \source : RTL code directory, contains all files pertaining to the FPGA design (top level file, clock generation, etc.). Separate subdirectory \source\ip contains Lattice IP cores with netlist and simulation files.

The following RTL Verilog files are contained in the \source directory:

- sdi_hdmi_top.v (top level RTL file)
- pcs_dvi_10b_diamond.v
- ecp3pcs.v
- sdi_core_ipexpress_bb.v
- mico32.v
- pll_96_23.v
- pll_100_96.v
- pll_148_5_27.v
- pll_100_148_5.v
- sdi_to_hdmi_ctrl.v
- yCbCr_to_RGB.v
- hdmi_transmitter_bb.v
- sdi_audio_params.v
- audio_enable_gen.v
- ch_status_gen.v
- sdi_audio_de_embedder.v

- aes3_packet_extr_bb.v
- i2s_encoder.v
- hdmi_audio_enc_bb.v
- fifo_34x16.v
- mult_9x9_UASB.v
- orcastra.v
- hdmi_to_sdi_ctrl.v
- RGB_to_YCbCr.v
- hdmi_receiver_bb.v
- hdmi_lnk_ctl.v
- rlos_reset.v
- resync_pulse.v
- sdi_audio_embedder.v
- i2s_decoder.v
- aes3_crc.v
- aes3_packet_gen_bb.v
- vpid_gen.v
- hdmi_audio_dec_bb.v
- i2s_enc.v
- iq_i2c_rom_top_vlk_bb.v
- pmi_ram_dp.v
- fifo_64x48.v

Programming the Design

The demonstration requires deployment of FPGA design usually to the SPI FLASH memory. This chapter will discuss downloading bitstream files (.bit) to the non-volatile memory using Diamond Programmer.

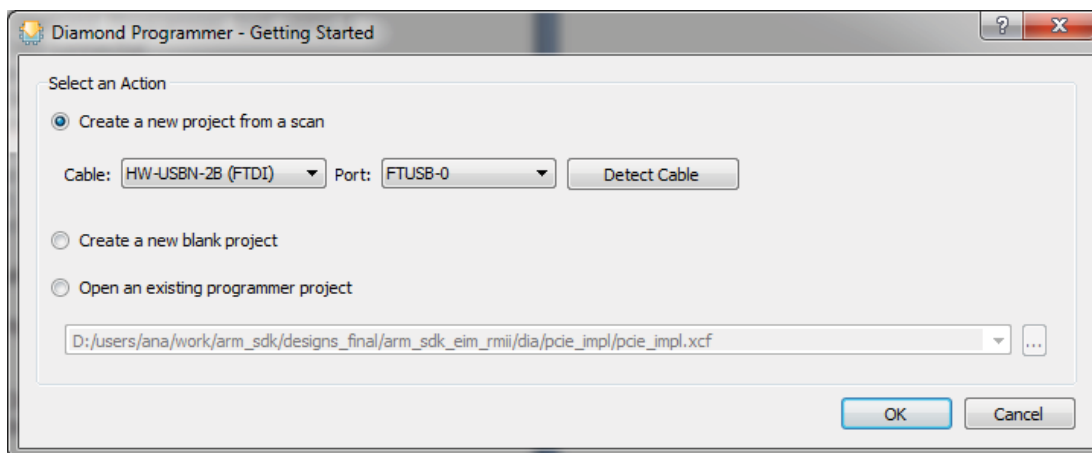
Deployment Using Diamond Programmer

SPI FLASH can be programmed using Lattice's Diamond Programmer tool, using a Lattice Programming Cable connected to the JTAG connector.

To download the bitstream file to the SPI FLASH on the board:

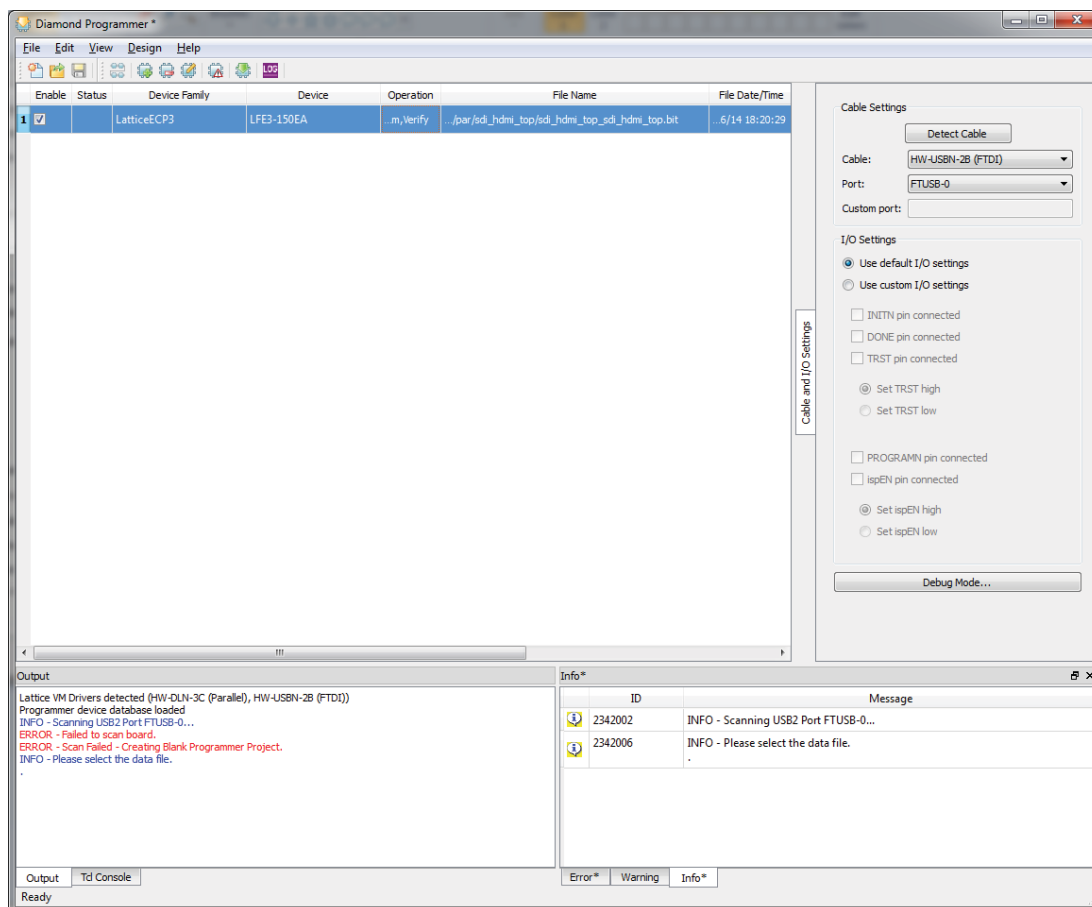
1. Remove any Lattice USB Programming cables from your system.
2. Connect the power supply to the Sparrowhawk FX board.
3. Connect a USB cable from your computer to the Sparrowhawk FX board. Give the computer a few seconds to detect the USB device.
4. In Diamond, choose **Tools > Programmer**.
5. In the Getting Started dialog box, choose **Create a new project from a scan**.
6. Click **Detect Cable**.
 - a. In the Cable box, select **HW-USBN-2B (FTDI)**.
 - b. In the Port box, choose **FTUSB-0**.
7. Click **OK**.

Figure 13. Getting Started Dialog Box in Diamond Programmer Tool



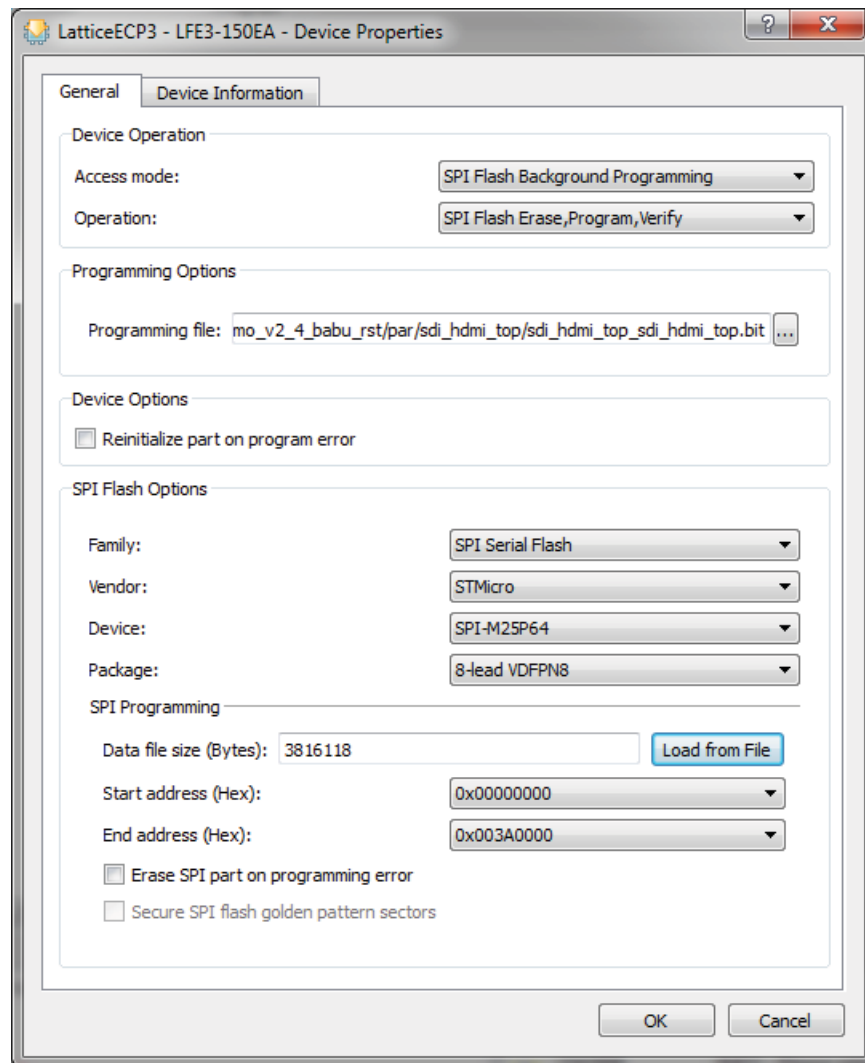
8. Programmer view is displayed in Diamond Programmer. Under the Device Family choose **LatticeECP3**.
9. Under the Device choose **LFE3-150EA**.

Figure 14. Programmer View in Diamond Programmer Tool



10. Double-click the **Operation** field.
 - a. Under the Access mode, choose **SPI Flash Background Programming**.
 - b. Under Operation, choose **SPI Flash Erase, Program, Verify**.
 - c. In the Programming Options, choose the programming file by browsing to the directory with .bit file you wish to download.
 - d. In the SPI Flash options choose **SPI Serial Flash, STMicro, SPI-M25P64, 8-lead VDFN8**.
 - e. Click **Load from File**.
 - f. Click **OK**.

Figure 15. Configuring SPI Flash Options in Diamond Programmer Tool



11. Click the **Program** button  on the Programmer toolbar to initiate the download.
12. If the programming process succeeded, you will see a green-shaded PASS in the Programmer Status column. Check the Programmer output console to see if the download passed.
13. Reboot the board by toggling the switch SW6.

Resource Utilization on LatticeECP3

The device resource utilization is summarized in Table 14.

Table 14. Resource Utilization for Different Modules

Modules	Slices	LUTs	Registers
sdi_to_hdmi_ctrl	117	64	145
tri_rate_sdi_phy_ip	1120	1509	1086
mico32	1511	2088	996
i2s_decoder	61	18	104
hdmi_transmitter	306	403	312
hdmi_audio_enc	1564	1575	1678
sdi_audio_embedder	597	698	736
sdi_audio_de_embedder	532	690	524
iq_i2c_rom	44	51	48
i2s_enc	45	63	33
hdmi_to_sdi_ctrl	264	263	237
hdmi_receiver	747	1049	722
hdmi_lnk_ctl	33	19	51
hdmi_audio_dec	287	202	449
Total	9471	14667	9355

Technical Support Assistance

Submit a technical support case via www.latticesemi.com/techsupport.

Revision History

Date	Version	Change Summary
June 2015	1.0	Initial release.